Reactive Statistical Mapping: Towards the Sketching of Performative Control with Data

Nicolas d'Alessandro[1], Joëlle Tilmanne[1], Maria Astrinaki[1], Thomas Hueber[2], Rasmus Dall[3], Thierry Ravet[1], Alexis Moinet[1], Huseyin Cakmak[1], Onur Babacan[1], Adela Barbulescu[2], Valentin Parfait[1], Victor Huguenin[1], Emine Sümeyye Kalaycı[1], and Qiong Hu[3]

[1]Numediart Institute for New Media Art Technology, University of Mons, Belgium
{nicolas.dalessandro,joelle.tilmanne,maria.astrinaki,thierry.ravet,alexis.moinet,huseyin.cakmak,onur.babacan}@umons.ac.be
[2]GIPSA-lab, UMR 5216/CNRS/INP/UJF/Stendhal University, Grenoble, France
{thomas.hueber,adela.barbulescu}@gipsa-lab.grenoble-inp.fr
[3]Centre for Speech Technology Research, University of Edinburgh, Scotland, UK
r.dall@sms.ed.ac.uk,qiong.hu@ed.ac.uk
http://www.numediart.org/hmapper

Abstract. This paper presents the results of our participation to the ninth eNTERFACE workshop on multimodal user interfaces. Our target for this workshop was to bring some technologies currently used in speech recognition and synthesis to a new level, i.e. being the core of a new HMM-based mapping system. The idea of statistical mapping has been investigated, more precisely how to use Gaussian Mixture Models and Hidden Markov Models for realtime and reactive generation of new trajectories from inputted labels and for realtime regression in a continuous-to-continuous use case. As a result, we have developed several proofs of concept, including an incremental speech synthesiser, a software for exploring stylistic spaces for gait and facial motion in realtime, a reactive audiovisual laughter and a prototype demonstrating the realtime reconstruction of lower body gait motion strictly from upper body motion, with conservation of the stylistic properties. This project has been the opportunity to formalise HMM-based mapping, integrate various of these innovations into the Mage library and explore the development of a realtime gesture recognition tool.

Keywords: Statistical Modelling, Hidden Markov Models, Motion Capture, Speech, Singing, Laughter, Realtime Systems, Mapping.

1 Introduction

The simulation of human communication modalities, such as speech, facial expression or body motion, has always been led by the challenge of making the virtual character look "more realistic". Behind this idea of realness, there are inherent properties that listeners or viewers have learnt to expect and track with great accuracy. Empirical studies, such as the Mori's vision for robotics [1], tend

to demonstrate that this quest for human likelihood is highly non-linear, encountering a well-known phenomenon called the *uncanny valley*, i.e. an unexpected shift from empathy to revulsion when the response of the virtual character has *"approached, but failed to attain, a lifelike appearance"* [2].

1.1 A Content-Oriented Approach of Expressivity

For the last decades, research fields like sound synthesis, computer graphics or computer animation have faced this issue of the uncanny valley in their own ways. However common trends can be highlighted. The primary goal in producing artificial human modalities has always been to "preserve the message", i.e. what is heard or seen is at least understood correctly. In speech synthesis, this is called *intelligibility* [3]. We can transpose this property to images or motion, as it refers to the readability of what is generated: a smile, a step, a laughter sound, etc. Later the target has evolved to "make it look more natural". This trend of *naturalness* is what has brought nearly everybody in these fields to use recordings of actual human performances, thus moving beyond explicit rule-based modelling. We can retrieve this concept in the development of non-uniform unit selection in speech synthesis [4], giga-sampling in music production [5], high-resolution face scanning [6] or full-body motion capture [7] in animation.

Nowadays the target is to bring *expressivity* and *liveliness* to these virtual characters. This idea goes further than naturalness as it expects the virtual character to display a wide range of convincing emotions, either automatically or by means of some authoring. For at least a decade, the typical approach to this problem has been to extend the amount and variability of the recorded data, hoping to gather enough utterances of the expected emotions so as to encompass what we perceive as human. If this idea is meaningful, one might conjecture that the way of representing and using this massive amount of data is not very lively nor flexible. For instance, non-uniform unit selection in speech synthesis concatenates very long utterances from the original recordings [4] and static or dynamic postures in animation are often blended from original sequences without a deep understanding of the production mechanisms.

1.2 Performative Control and Machine Learning

Although the use of a large amount of data has clearly improved some aspects of virtual human-like character animation[1], we could argue from the current results that this approach on its own has not been able to fully climb the uncanny valley. Indeed speech synthesis and face/body animation resulting from monolithic transformations of original recordings keep having something inappropriate and

[1] Our approach is really transversal to all the virtual character's modalities: voice, face and body. Therefore we tend to use the terms "rendering" and "animation" for both sounds and visuals. This distinction is rarely done for sound, as "synthesis" is often used for the whole process. Though a similar nuance can be found in speech, when respectively referring to segmental and suprasegmental qualities [3].

confusing [8]. In this research, we think that user interaction has a great role to play in how a given virtual human-like character will be considered more expressive and lively. Indeed we agree that expressivity is a matter of displaying a great variability in the rendered output, but we think that these variations need to be contextually relevant. This "context" is a sum of elements surrounding the virtual character at a given time and user interaction has a big impact on how the rendering system should behave. Our way of reading the Mori's law is that a significant part of the perceived humanness sits in very subtle details. In this work, we will focus on how the animation trajectories can reactively adapt to the user interaction on the very short term, i.e. within milliseconds. We think that there is a lack in the literature about how these trajectories should react according to very short-term gestures, as most of the research is focusing on a larger time window and the overall discussion of dialog systems.

Our approach towards bringing together large multimodal datasets and short-term user interaction is to use machine learning. There is already a very large body of literature on applying machine learning techniques in the fields of gesture or voice recognition [9], gesture or voice synthesis [10], gesture or voice conversion [11], and what is called *implicit mapping*, i.e. the use of statistical layers as a way of connecting inputs to outputs in interactive systems [12]. Multimodal animation of virtual characters brings two main constraints to take into account when it comes to statistical modelling. On the one hand, dimensionality of the data is very big (see Section 3 for details). On the other hand, we aim at creating animation *trajectories*, which means that the temporal quality of the generated outputs is crucial. Besides these specific animation-related constraints, we want to enable short-term interaction with the virtual character. Therefore the way of handling statistics in our system requires to be fully *realtime*.

1.3 A New Framework for GMM/HMM-Based Mapping

In this project, we have decided to investigate the use of Gaussian Mixture Modelling (GMM) and Hidden Markov Modelling (HMM) as statistical tools to abstract big collections of multimodal data, use such knowledge to animate virtual characters in realtime and enable various kinds of user interactions to happen. GMM/HMM offers great ability to cover complex feature spaces and HMM is designed for taking care of the temporal structure of trajectories to be rendered. Moreover the development of GMM/HMM-based machine learning techniques has been greatly boosted by recent advances in speech technology research. Particularly the idea of HMM-based speech synthesis has emerged in the last decade [10]. A synthesis system called HTS has become a reference in this field [13]. There are many pieces of innovative research surrounding GMM/HMM-based generation. Three of them have particularly encouraged our team to envision a new GMM/HMM-based framework for virtual character animation:

- the adaptation of the speech algorithms to motion capture data [14, 15];
- the modification of the core generation algorithms to be fully realtime [16];
- the integration of mapping functions inside the HMM framework [17, 18].

As a result, we have decided to create a new framework that accepts and decodes user interaction gestures in realtime, finds the appropriate statistical context to apply mapping strategies and is able to generate new trajectories to partially or totally animate a virtual character. In this work, this prototype framework has been tested for a great amount of use cases, encompassing many modalities: speech, singing, laughter, face and body motion.

1.4 Outline of the Paper

In this paper, we will first present a more detailed background theory on the statistical models that are used in our system (cf. Section 2). In Section 3 we describe the particularities of the datasets that we have been using: speech, laughter, singing, facial and gait motion. Section 4 explains the various use cases in which we enabled the realtime trajectory generation. Section 5 focuses on use cases where mapping is the fundamental feature. Further details on the architecture are given in Section 6. Finally we conclude in Section 7.

2 Statistical Feature Mapping: Theoretical Aspects of GMM-Based and HMM-Based Mapping Techniques

The research described in this paper relies on a very specific use of machine learning techniques based on Gaussian Mixture Models (GMMs) and Hidden Markov Models (HMMs). The aim of this Section is to describe a set of background theories that are necessary to understand the following Sections. In the next parts, we first recall the theoretical context of *statistical feature mapping*. Then we give an overview on how to turn GMM and HMM into mapping layers. We also give more details on how to generate the HMM-based trajectories in realtime, as it is required for building an interactive system.

2.1 Statistical Feature Mapping

The problem of *feature mapping* refers to the prediction of a vector of target variables $\hat{\mathbf{y}}$ (also called target *features*), from the observation of an unseen vector of input variables $\mathbf{x}$. This problem can be divided into three categories:

- *regression*, also referred to as *continuous mapping*, i.e. the case where both $\mathbf{y}$ and $\mathbf{x}$ will comprise continuous variables;
- *classification*, i.e. the case where $\mathbf{y}$ will represent a discrete set of class labels while $\mathbf{x}$ will comprise continuous variables;
- *generation*, also referred to as *synthesis*, i.e. the case where $\mathbf{x}$ will represent class labels and $\mathbf{y}$ the continuous variables we want to estimate.

The feature mapping problem can be viewed from a probabilistic viewpoint. It consists in finding the vector of target variables $\hat{\mathbf{y}}$ which maximises the conditional probability $p(\mathbf{y}|\mathbf{x})$ of $\mathbf{y}$ given $\mathbf{x}$, such as:

$$\hat{\mathbf{y}} = \arg\max_{\mathbf{y}} \{\ p(\mathbf{y}|\mathbf{x})\ \} \tag{1}$$

In *supervised* machine learning, this conditional probability is estimated during the training phase from a dataset $\mathbf{x} = [\mathbf{x_1}, ..., \mathbf{x_N}]$ comprising N observations of $\mathbf{x}$ together with the corresponding observations of the values of $\mathbf{y} = [\mathbf{y_1}, ..., \mathbf{y_N}]$. Two types of approaches can be envisioned to estimate this conditional probability: the discriminative or the generative approach.

In a *discriminative* approach, this conditional probability distribution (also referred to as the posterior probability distribution) is estimated directly from the training data. In other words, such an approach provides a model only for the target variables conditional on the observed variables and does not provide a complete probabilistic model of all the variables (observed and hidden ones). Examples of discriminative models include Linear Discriminant Analysis (LDA), Conditional Random Fields (CRF), Artificial Neural Networks (ANN), etc.

In a *generative* approach, the conditional probability is not estimated directly, but derived from the joint probability distribution $p(\mathbf{x}, \mathbf{y})$ using Bayes' theorem:

$$p(\mathbf{x}, \mathbf{y}) = \mathbf{p}(\mathbf{y}|\mathbf{x})\mathbf{p}(\mathbf{x}) = \mathbf{p}(\mathbf{x}|\mathbf{y})\mathbf{p}(\mathbf{y}) \tag{2}$$

where $p(\mathbf{x}|\mathbf{y})$ is the likelihood function, $p(\mathbf{y})$ the prior probability – i.e. the probability of $\mathbf{y}$ *before* seeing $\mathbf{x}$ – and $p(\mathbf{x})$ is usually viewed as a normalisation constant. Note that the joint probability $p(\mathbf{x}, \mathbf{y})$ can be either modelled explicitly or implicitly via the separate estimation of the likelihood function and the prior probability. Examples of generative models includes GMMs and HMMs.

Using Generative Models Generative and discriminative approaches have their own advantages and drawbacks. An extensive discussion about which approaches would be more suitable for addressing a specific application is far beyond the scope of this paper. However we focused on generative models for two main reasons.

The first advantage of the generative approach is that the inclusion of *prior knowledge* arises naturally. This may be extremely convenient to address problems, considered as ill-posed, for which there is no clear one-to-one mapping between input and target feature spaces. Furthermore, as prior probabilities do not depend on the input observation $\mathbf{x}$, they may be estimated on a much larger dataset than the training set. Prior knowledges can be used to constrain the mapping process to generate acceptable outputs (as in speech recognition, where a language model gives the probability of having a specific sequence of words in a specific language, independently from the observed acoustic signal).

The second advantage is its *flexibility*. Estimating the joint probability distribution over input and target variables $p(\mathbf{x}, \mathbf{y})$ allows to address several mapping problems with the same model. As an example, let us consider a mapping problem between a continuous feature space $\mathbf{x}$ and a discrete space of k classes C_k. The same approach which aims at estimating $p(\mathbf{x}, \mathbf{C_k})$ could be used to address both the related classification problem by deriving the conditional probability

$p(C_k|\mathbf{x})$, and the symmetrical problem of trajectory generation, by sampling the conditional probability $p(\mathbf{x}|\mathbf{C_k})$ – i.e. the likelihood function.

However we have to keep in mind that generative approaches are known to require a lot of training data. In fact, the dimensionality of both input $\mathbf{x}$ and target observations $\mathbf{y}$ may be high, and consequently a large training set is needed in order to be able to determine $p(\mathbf{x}, \mathbf{y})$ accurately.

In this project, we focused on two generative approaches, which are GMMs and HMMs. The theoretical aspects of these two techniques are mainly presented in the context of *regression*, plus a specific focus on *realtime generation* and the MAGE framework. Indeed more general aspects of GMM/HMM-based classification [9] and generation [10] have already been extensively described.

From now on, sequences of input and target feature vectors are noted respectively $\mathbf{x}$ and $\mathbf{y}$, and are defined as: $\mathbf{x} = [\mathbf{x}_1, .., \mathbf{x}_t, ..., \mathbf{x}_T]$ and $\mathbf{y} = [\mathbf{y}_1, .., \mathbf{y}_t, ..., \mathbf{y}_T]$, where $\mathbf{x}_t$ are $\mathbf{y}_t$ are respectively D_x and D_y dimensional vectors of input and target features observed at time t (T is the sequence length).

2.2 GMM-Based Mapping

During the training phase, the joint probability density function (pdf) of input and target features is modelled by a GMM such as:

$$p(\mathbf{z}|\lambda) = p(\mathbf{x}, \mathbf{y}) = \sum_{m=1}^{M} \alpha_m N(\mathbf{z}, \mu_m, \Sigma_m) \tag{3}$$

with

$$\mathbf{z} = [\mathbf{x}, \mathbf{y}],\ \mu_m = \begin{bmatrix} \mu_m^{\mathbf{x}} \\ \mu_m^{\mathbf{y}} \end{bmatrix},\ \Sigma_m = \begin{bmatrix} \Sigma_m^{\mathbf{xx}} & \Sigma_m^{\mathbf{xy}} \\ \Sigma_m^{\mathbf{yx}} & \Sigma_m^{\mathbf{yy}} \end{bmatrix} \tag{4}$$

where λ is the parameter set of the model, $N(., \mu, \Sigma)$ is a normal distribution with mean μ and covariance matrix Σ, M is the number of mixture components, and α_m is the weight associated with the m^{th} mixture component (prior probabilities). Given a training dataset of input and target feature vectors, the parameters of a GMM (weights, mean vectors and covariance matrices for each component) are usually estimated using the expectation-maximisation (EM) algorithm. The initial clustering of the training set can usually be obtained using the k-means algorithm.

In the mapping phase, a conditional pdf $p(\mathbf{y}_t|\mathbf{x}_t, \lambda)$ is derived, for each frame t, from the joint pdf $p(\mathbf{x}_t, \mathbf{y}_t)$ estimated during training, such as described in Eq. 5 to 9. The mathematical basis of this derivation can be found in [19].

$$p(\mathbf{y}_t|\mathbf{x}_t, \lambda^{[\mathbf{xy}]}) = \sum_{m=1}^{M} p(c_m|\mathbf{x}_t) p(\mathbf{y}_t|\mathbf{x}_t, m, \lambda^{[\mathbf{xy}]}) \tag{5}$$

where λ is the model parameter set. The posterior probability $P(c_m|\mathbf{x_t})$ of the class c_m given the input vector $\mathbf{x_t}$, and the mean and covariance of the class-dependent conditional probability $P(\mathbf{y}_t|\mathbf{x}_t, m, \lambda^{[\mathbf{xy}]})$ are defined as:

$$p(c_m|\mathbf{x_t}) = \frac{\alpha_m N(\mathbf{x_t}, \mu_m^{\mathbf{x}}, \sum_m^{\mathbf{xx}})}{\sum_{p=1}^{M} \alpha_p N(\mathbf{x}, \mu_p^{\mathbf{x}}, \sum_p^{\mathbf{xx}})} \tag{6}$$

and

$$p(\mathbf{y}_t|\mathbf{x}_t, m, \lambda^{[\mathbf{xy}]}) = N(\mathbf{y}_t, E_{(m,t)}, D_{(m,t)}) \tag{7}$$

with

$$E_{(m,t)} = \mu_m^{\mathbf{y}} + \Sigma_m^{\mathbf{yx}} {\Sigma_m^{\mathbf{xx}}}^{-1} (\mathbf{x}_t - \mu_m^{\mathbf{x}}) \tag{8}$$

$$D_{(m,t)} = \Sigma_m^{\mathbf{yy}} - \Sigma_m^{\mathbf{yx}} {\Sigma_m^{\mathbf{xx}}}^{-1} \Sigma_m^{\mathbf{xy}} \tag{9}$$

Two approaches can then be envisioned to address a regression problem with a GMM. In the first one, referred here to as the *MMSE-GMR* for "Gaussian Mixture Regression based on the Minimum Mean Square Error Criterion", the target feature vector $\hat{\mathbf{y}}_\mathbf{t}$ estimated from the given source vector $\mathbf{x_t}$ observed at time t, is defined as $\hat{\mathbf{y}}_t = E[\mathbf{y}_t|\mathbf{x}_t]$ (where E means expectation) and can be calculated by solving Eq. 10. In particular, this approach has been described in [20] and [21] in the context of statistical voice conversion.

$$\hat{\mathbf{y}}_t = \sum_{m=1}^{M} p(c_m|\mathbf{x}_t) E_{m,t}^{\mathbf{y}} \tag{10}$$

The second approach, proposed by Toda et al. [22], is here referred to as *MLE-GMR*, for "Gaussian Mixture Regression based on Maximum Likelihood Criterion". The target feature vector $\hat{\mathbf{y}}_\mathbf{t}$ is defined as the one which maximises the likelihood function such as:

$$\hat{\mathbf{y}}_t = \arg\max_{\mathbf{y_t}} \{\ p(\mathbf{y}_t|\mathbf{x}_t, \lambda^{[\mathbf{xy}]})\ \} \tag{11}$$

and can be estimated by solving Eq. 12:

$$\hat{\mathbf{y}}_t = \left[\sum_{m=1}^{M} p(c_m|\mathbf{x}_t) D_{(m,t)}^{\mathbf{y}}\right]^{-1} \left[\sum_{m=1}^{M} p(c_m|\mathbf{x}_t) D_{(m,t)}^{\mathbf{y}} E_{(m,t)}^{\mathbf{y}}\right] \tag{12}$$

Trajectory GMM The MLE-GMR approach is commonly combined with a constraint on the smoothness of the predicted trajectories. The GMM is then referred to as a *trajectory GMM*. In that case, each target feature vector $\mathbf{y_t}$ of the training set is augmented by its N-order derivatives such as $\mathbf{Y}_t = [\mathbf{y}_t\ \Delta\mathbf{y}_t]$ (the method is here presented with $N = 1$). The joint probability distribution $p(\mathbf{Y}, \mathbf{x})$ is modelled similarly to the MLE-GMR approach. However the mapping process is done differently. The sequence of target feature vectors is not

estimated on a frame-by-frame basis as in previous approaches, but rather "all-at-once". The estimated sequence of target feature vectors is defined as the one that maximises the likelihood of the model, with respect to the continuity of its first N derivatives:

$$\hat{\mathbf{Y}} = \arg\max_{\mathbf{Y}} \{\ p(\mathbf{Y}|\mathbf{x}, \lambda^{[\mathbf{xY}]})\ \} \tag{13}$$

which can be estimated by solving the closed-form equation in Eq. 14:

$$\hat{\mathbf{y}} = \left(W^T D_{\hat{m}}^{-1} W\right)^{-1} W^T D_{\hat{m}}^{-1} E_{\hat{m}} \tag{14}$$

where W is a $[2D_xT \times D_yT]$ matrix representing the relationship between static and dynamic feature vectors (Fig. 1) and $\hat{m} = [\hat{m}_1, .., \hat{m}_t, ..., \hat{m}_T]$ is the sub-optimum sequence of mixture components defined as:

$$\hat{m} = \arg\max_{m} \{\ p(c_m|\mathbf{x}, \lambda)\ \} \tag{15}$$

and determined using the Viterbi algorithm (in our experiment, and similarly to what was reported in [22], similar results were obtained using a forward-backward approach which takes into account in a probabilistic manner the contributions of all mixture components).

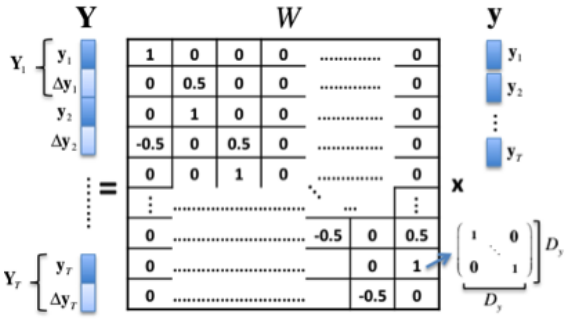

Fig. 1. W is a $[2D_xT \times D_yT]$ matrix representing the relationship between static and dynamic feature vectors. It is used in the computation of output trajectories of Eq. 14.

2.3 Realtime Trajectory Generation

If the "all-at-once" approach suggested by Eq. 14 can guarantee smooth trajectories, it prevents these trajectories to be generated in realtime, and therefore the system to be interactive. In Section 1, we have claimed that interactivity was a required property for our system, in order to produce expressive virtual characters. For the last few years, Astrinaki et al. have worked to find a new trade-off between the smoothness and the system-wise reactiveness of generated

trajectories [16, 17]. The idea of finding the Maximum Likelihood (MLE) over the whole sequence as required to fill the whole W matrix in Eq. 14 has been replaced by finding a sub-optimal version of the ML on a sliding window. This new algorithm, called *Short-Term Maximum Likelihood Parameter Generation (ST-MLPG)*, enables the trajectory generation process to start when the system receives the first class label and not at the end of a full sequence of class labels.

The ST-MLPG is a key feature of the open-source library called MAGE [23]. The MAGE software actually enables to create interactive systems that generate smooth trajectories in realtime, which is the main reason why this software is the trajectory generation engine in this project. However the MAGE library was very tied to the HMM-based approach – explained in the next part – and the HTS architecture when we started the project, which explains why we have significantly modified its architecture along the way (see 6 for details).

2.4 HMM-Based Mapping

Hidden Markov Modelling has been used for a long time in *temporal pattern recognition*, as for instance in automatic speech recognition (ASR), handwriting, or gesture recognition. More recently HMM has also been successfully used for parameter generation, such as in HMM-based speech synthesis [10]. HMM can be considered as a generalisation of GMM where the latent variables – i.e. the hidden states – are derived from a Markov process rather than being independent from each other. Latent variables are controlling the mixture component to be selected for each observation. As HMM aims at representing explicitly the temporal evolution of features, it is adapted to model data that can be clustered not only by its distribution but also by its temporal evolution.

Due to this temporal structure, achieving a regression task (or mapping) with a HMM requires a more complex framework. The algorithm proposed in [18] combines a HMM-based classification and a HMM-based parameter generation, from the same trained models. The mapping operation can therefore happen within the HMM currently in use and "passed" from the classification to the generation operations. This process is illustrated in Fig. 2. We traditionally assume that sequences of feature vectors which constitutes the training set are temporally segmented and labeled. This segmentation can be obtained by annotating the data, either manually, or by using an initial set of already-trained HMMs and a forced-alignment procedure.

In the training phase, sequences of input and target feature vectors (completed with their first N derivatives) are modelled jointly, for each class, by a "full-covariance" HMM, i.e. an HMM for which the emission probability distribution is modelled, for each state q, by a normal distribution with a full-covariance matrix, as defined by Eq. 3 (with $M = 1$). After initialisation, models are typically trained using the following standard procedure: models are first trained separately, using the standard Baum-Welch re-estimation algorithm and then processed simultaneously, using an embedded training strategy. Since input/target features are often sensitive to context effects (for instance, co-articulation and anticipation in speech), context-dependency is often introduced in the modelling.

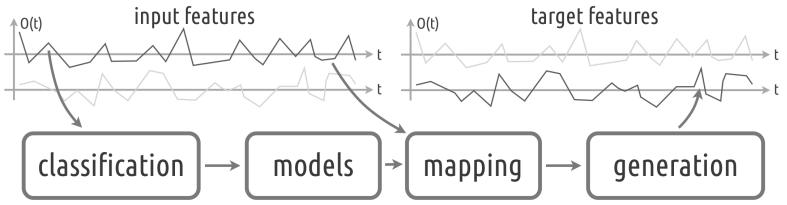

Fig. 2. Summary of the framework required to achieve a HMM-based mapping: HMM-based classification of input features, query of models based on obtained class labels, mapping routine from models and input features and generation of target features

Context-dependent models are created by adding information about left and right contexts to the initial models. A tree-based state-tying technique is then eventually used to tackle the problem of data sparsity (context-dependent models having only a few occurrences in the training dataset).

In the mapping phase, the sequence of target feature vectors $\hat{\mathbf{y}}$ is estimated from the sequence of input feature vectors $\mathbf{x}$ such as:

$$\hat{\mathbf{y}} = \arg\max_{\mathbf{y}} \{\ p(\mathbf{y}|\mathbf{x})\ \} \tag{16}$$

with

$$p(\mathbf{y}|\mathbf{x}, \lambda) = p(\mathbf{Y}|\lambda, q, \mathbf{x}).p(\lambda, q|\mathbf{x}) \tag{17}$$

with $Y = Wy$ (see Fig. 1), λ the parameters set of the HMM and q the HMM state sequence. Eq. 17 is the product of two conditional probability terms which can be maximised separately:

1. $p(\lambda, q|\mathbf{x})$ which is related to the *classification stage* which aims at estimating the most likely HMM (or sequence of concatenated HMMs) $\hat{\lambda}$ with the corresponding sequence of states $\hat{q}$ such as:

$$(\hat{\lambda}, \hat{q}) = \arg\max_{(\lambda,q)} \{\ p(\lambda, q|\mathbf{x})\ \} \tag{18}$$

   using the Viterbi algorithm. Using Bayes' theorem, we obtain $p(\lambda, q|\mathbf{x}) = p(\mathbf{x}|\lambda, q).p(\lambda, q)$ and see that this classification stage allows the introduction of external knowledges, via the use of prior probabilities on class sequences, which could be used to constrain the mapping (in speech recognition, this term would be related to the language model).
2. $p(\mathbf{y}|\lambda, q, \mathbf{x})$ which is related to the *synthesis stage* and could be maximised similarly to as the GMM-based mapping technique, using Eq. 14. This is similar to the trajectory GMM technique, but here a continuity constrain is imposed on the estimated feature trajectories. The corresponding HMM is thus often referred to as a *trajectory HMM*. The modification proposed in the short-term MLPG can also be applied at this level.

2.5 Classification vs. Mapping: A Modelling Trade-Off

The integration of HMM-based classification, mapping and generation within the same framework gives us a chance to discuss about a specific modelling trade-off. Indeed from the same feature space (input and target features), various modelling strategies can be applied. On the one hand, we can create a large amount of small clusters. In that case, the labelling is very precise, the classification task needs to be very discriminative and the parameter generation has a very narrow area from which to get its target values, limiting the influence of the mapping. On the other hand, we can create a small amount of large clusters. In that case, the labelling is looser, discrimination between classes is easier and the parameter generation has a wide area from which to get its target values, hence much more relying on the mapping to browse the subspace. A summary of this trade-off is given in Fig. 3. The two approaches will lead to very different kinds of applications.

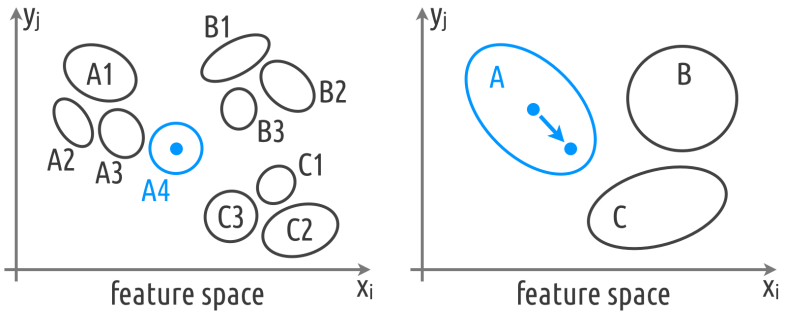

Fig. 3. Explanation of the modelling trade-off between classification, mapping and generation. We can choose between many small clusters where mapping is limited or few big clusters where mapping is primordial to browse the subspace.

3 Description of Data Types Used for Modelling

One main objective of this project was to design and develop a framework where GMMs and HMMs can be applied to a very great variety of data types. In this Section, we give a more exhaustive description of the data types that we have addressed and the databases that we have used. It gives a solid ground for understanding the feature spaces that we are manipulating.

3.1 Speech

The voice models that we used for speech are either identical or similar to the ones found in the HTS software [13]. Indeed our work with the speech databases has mainly used standard HTS training but it has also aimed at retraining some

voices, such as the ones used by the incremental speech synthesis application detailed in Section 4.1. For practical reasons, during development we generally work with the standard voice model from HTS demo scripts which is SLT, an American English female speaker from the CMU ARCTIC database [24], sampled at 48 kHz, 16-bit PCM. The training parameters for these models are the default ones in HTS for a sampling rate of 48 kHz. The input audio signals are split into 1200-sample long frames with a frame shift of 240 samples. For each frame, a parametric representation is extracted. The vocal tract is modelled by a 35-order MLSA filter [25] whose parameters are $\alpha = 0.55$ and $\gamma = 0$, whereas the glottal source is described by a voiced/unvoiced decision and a pitch value, if any. On a higher level, each file from the training dataset has a phonetic transcription, from which phoneme duration models are trained, as well as additional information about syllables, words, sentences, accentuations, etc. [26]. For each parameter, this contextual information is used to train binary decision trees whose leaves are Gaussian models of the parameter.

3.2 Singing

Many similarities exist between speech and singing voice, though the differences are significant, especially for analysis and modelling. Some of the more prominent phenomena specific (though not necessarily exclusive) to Western classical singing[2] in contrast to regular speech are the higher voiced/unvoiced ratio, vibrato, higher range of loudness and pitch, singer's formant and modification of vowels at high pitches [27]. These and other differences between speech and singing voice lead to some challenges in analysis modelling of singing.

The assumption of a decoupled source-filter is reasonably accurate especially for speech, but source-filter interactions are known to occur in various situations. For example, coupling effects increase by high pitch or high-impedance vowels like [*u*] [28]. Another challenge is that speech analysis algorithms such as pitch or *Glottal Closure Instant (GCI)* estimators often do not work as intended and this results in loss of performance [29, 30]. Finally prosody models based on linguistic and phonetic information are almost never applicable due to the nature of singing. Instead different prosody models that take musical gestures and score information into account may be necessary.

In order to capture these singing-specific stylistics, we designed and recorded a new singing database for our purposes. The main challenge was to define the stylistic space in a meaningful manner. After consulting different systems of categorising singing voice, we decided that our approach is to set a *cognitive target* for the singer. Consequently we only specified more abstract stylistic information in the database design and aimed to capture the singing phenomena as they occur naturally. We contracted a professional female singer, EB, and the recordings were done in a professional recording studio. Seven pieces were recorded with multiple renditions in three different styles: *neutral*, *classical* and

[2] In our work we constrain our domain to Western classical singing, because it is well-formalised and a sizeable amount of previous technical work exists.

belting. Since naturalness was high priority, the pieces were chosen from EB's repertoire of jazz, classical, folk and contemporary music, which she was comfortable and confident in performing. A limited amount of pure-vowel recordings were also done, consisting of some selected pieces and specific pitch gestures such as flat and constant ascent/descent. Contemporaneous electroglottography (EGG) recordings were also made, in order to establish ground truth for pitch and enable GCI-synchronous analysis methods. The resulting database is more than 70 minutes long, containing a rich variety of singing samples.

3.3 Audio-Visual Laughter

For addressing laughter modelling, the AV-LASYN database was used. The AV-LASYN database is a synchronous audio-visual laughter database built for the purpose of audio-visual laughter synthesis. The next paragraphs give an overview of the recording pipeline. Audio data was recorded at high sampling rate (96kHz) for eventual study of the impact of sampling rate on audio synthesis results. However, since it is a higher sampling rate than what common applications and research such as the present work need, we downsampled to 44.1kHz. Visual laughter was recorded using a marker-based motion capture system commercially available and known as OptiTrack. A set of 6 infrared cameras were used to track at 100 fps the motion of 37 markers glued on the subject. A seventh camera was used to record a grayscale video synchronised with all others. Among the 37 tracked markers are 4 markers placed on a headband. These helped to extract head motion from face deformation and make both available independently. After this separation process, we end up with 3 values for each facial marker (xyz coordinates) which corresponds to 99 values at each frame as well as 6 values at each frame that represent head motion (xyz coordinates and rotations around the same axes). This makes a 105-dimensional vector to represent overall face motion for a given frame. Neutral pose and the whole set of data corresponding to visual motion have been saved in the Biovision Hierarchy (BVH) format . The final corpus is composed of 251 segmented laughs. This corresponds roughly to 48 minutes of visual laughter and 13 minutes of audible laughter. For each laugh, the corpus contains: an audio file [44.1kHz, 16 bits], a BVH motion file that can be loaded in common 3D software (with the neutral pose, 6 channels for head motion, 3 channels for each of 33 facial markers), a binary motion file containing the same data as in the BVH to make it easier to load, a HTK label file containing phonetic transcriptions and temporal borders for each laughter phone.

3.4 Audio-Visual Affective Speech

Experiments were conducted on the BIWI 3D Audiovisual Corpus of Affective Communication [40] comprising a total of 1109 sentences (4.67 seconds long on average) uttered by 14 native English speakers (6 males and 8 females). The dense dynamic face scans were acquired at 25 frames per second by a realtime 3D scanner and the voice signal was captured by a professional microphone at a sampling rate of 16kHz. For the voice signals, fundamental frequency, signal

intensity and segment duration are also provided. Along with the detailed 3D geometry and texture of the performances, sequences of 3D meshes are provided, with full spatial and temporal matching across all sequences and speakers. For each speaker 80 utterances are recorded, half in a personal speaking style and half in an "emotional" manner, as they are asked to imitate an original version of the performance.

3.5 Stylistic Gait

In this work, we also used the Mockey database [31] as our stylistic gait motion capture database. This database was recorded using a commercial inertial motion capture suit called IGS-190, from Animazoo [32]. This motion capture suit contains 18 inertial sensors, which record the angles between "body segments" corresponding to a simplified human skeleton representation. The output of the motion capture suit are these angles, expressed in the Euler angle parameterisation, and the calculated 3D position of the root (hips), which is computed given the angles and the lengths of the leg segments. In the database, the walk of a professional actor impersonating different expressive styles was recorded. The eleven styles represented in the database were arbitrarily chosen for their recognisable influence on walk motions. These styles are the following: proud, decided, sad, cat-walk, drunk, cool, afraid, tiptoeing, heavy, in a hurry, manly. Each walk style is represented by a different number of steps in the database, ranging from 33 to 80 steps. Fig 4 gives an overview of these walking styles.

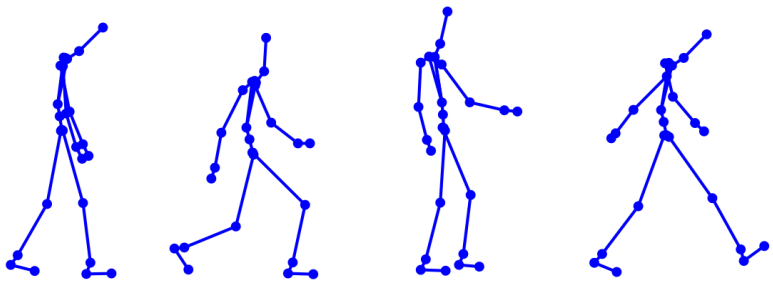

Fig. 4. Generic skeleton frames extracted from the Mockey database [31] and corresponding to different styles of gait, here: sad, afraid, drunk and decided

The Mockey mocap sessions are recorded in the Biovision Hierarchy (BVH) format. The skeleton from the Animazoo software is defined by 20 body segments, and each data frame hence contains 66 values. Three of the segments are in fact only added to make the simplified skeleton look closer to a real skeleton but have no degree of freedom in the motion capture. There are hence finally 57 values to analyse and model in our data, among which 54 joint angle values and 3 values for the skeleton root Cartesian coordinates. Since the Cartesian

coordinates of the root are calculated a posteriori from the joint angles, we only took into account the 54 joint angle values in our models. Furthermore, since the Euler angle representation is seldom optimal, we converted it to the exponential map angle parameterisation, which is locally linear and where singularities can be avoided. The Mockey database has been recorded at a rate of 30 frames per second. The walk sequences have been automatically segmented into left and right steps, based on the evolution of the hip joints angles.

4 Realtime and Reactive HMM-Based Generation

The first step in creating our new framework was to validate the adaptation of the realtime HMM-based parameter generation – such as described in Section 2 and implemented in MAGE – to the new data types described in Section 3. It brought us to implement a series of prototypes that are described in this Section.

4.1 Incremental Speech Synthesis

In several applications of Text-To-Speech (TTS) it is desirable for the output to be created incrementally. Such applications include reactive dialogue systems and speech devices for disabled people. However current TTS systems rely on full pre-specified utterances provided before the synthesis process begins, severely limiting the reactivity and realtime use of speech synthesis. While MAGE is capable of realtime synthesis, it is reliant on linguistic context labels which are computed offline prior to synthesis. This means the utterance to be synthesised is fixed and cannot be changed at run-time except to other pre-computed context labels. There is, however, nothing to prevent MAGE from synthesising from an incrementally created set of labels. Hence a new realtime linguistic front-end was created in order to allow for continuous incremental creation of the linguistic context labels for synthesis. A new front-end was chosen as current front-ends are simply nowhere near fast enough for realtime analysis – e.g. Festival takes over 1000ms to process an utterance (even a single word) and MARY-tts slightly above 200ms [33] – and they assume the full utterance is present at analysis time.

Linguistic Analysis The standard full-context labels used by the HTS engine includes a large amount of varying contexts used in the decision tree context clustering process. Many of these do not lend themselves to incremental processing as they rely on the presence of the full utterance. Therefore a reduced context set was decided upon based on the standard HTS set [34]. Any contexts related to the full utterance or phrases were removed as these are not available. Contexts regarding future syllables are included 'if available' and the system requires two future phones. The decision to retain two future phones, making the system in effect lagging two phones behind realtime, was made as informally listening to the speech when no future phones were included resulted in a significant degradation of the speech intelligibility. The resulting context labels included 23 contexts down from 53. Informally no noticeable drop in quality was perceived

on a voice re-trained on the reduced context set compared to a voice trained on the standard HTS contexts. The system works with word-sized chunks, such that every time a user inputs a complete word the system will provide the labels necessary to synthesise the word, with a minimum of two phonemes. The words are looked up in the CMUDict 0.4 dictionary from which stress patterns, syllables and phones are retrieved and the labels created. No letter-to-sound rules are included. If a word is not be in the dictionary, a filled pause is introduced instead ("uh").

Typing Interface A simple typing interface was implemented which allows a user to type in the string to be synthesised. It is however incredibly difficult to type as fast as the synthesis speed, so synthesis was slowed by a factor of 2.5 to allow a skilled typer to type fast enough. To further enhance the ability of the typist to type quickly, simple word prediction was added allowing the user to press the F1 to F5-keys to instantly insert a word predicted by the system.

4.2 Reactive Control of Audio and Visual Laughter

In this part we explain how the modelling of the laughter has been approached with HMMs, both for the sound and the face motion. Then we give a first description on how we turned these two processes into an interactive application.

Reactive Acoustic Laughter Synthesis HMM-based acoustic laughter synthesis is a problem that has been addressed recently [35, 36]. The same pipeline has been applied in this work to train an acoustic model of laughter using the AV-LASYN database described in Section 3.3. We have extracted 35 Mel-cepstral coefficients and log F0 as features to represent the acoustic signal. We used STRAIGHT [37] for this extraction process. Then, five-state, left-to-right HMMs were trained to model a set of laughter phones (see [35] for more information). From the synthesised F0, an excitation signal was derived and modified by DSM [38]. Finally synthesised MFCC trajectories and modified excitation signal were used to feed a source-filter model and generate the corresponding waveforms. With the acoustic models obtained as explained above, we were able to integrate acoustic laughter into Mage. Although there is still room for improvements, we have shown that reactive acoustic laughter synthesis is feasible through Mage. Further investigation is needed to have a better understanding of the behaviour of Mage for the synthesis of the specific signal of laughter, and this work will serve as a basis for further studies.

Visual Laughter Synthesis A similar process has been applied to visual data contained in the AV-LASYN database. As a reminder of Section 3.3, the visual data consists of facial points trajectories. Thirty-three markers on the face represented at each frame by 3 spatial coordinates are used. Six other values represent the head motion. The features that we used to train the HMMs were obtained

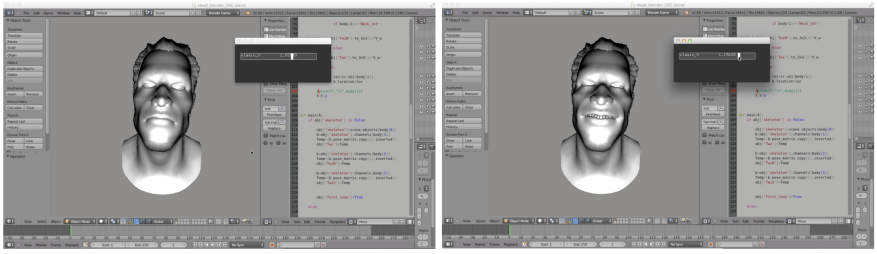

Fig. 5. Illustration of the reactive control of laughter intensity by having a MAGE application to send trajectories to the 3D face model in Blender through OSC

as follows. First we subtracted the neutral face from each frame so that the data represents only the facial deformation. Then a PCA analysis was performed and showed that 97% of the variability in the data is contained in the 4 first principal components. We hence decided to reduce the dimensionality of the data to 4. However the PCA analysis did not include the 3 values representing the head rotation for a matter of consistency of the data on which PCA was applied. We thus end with a 7 dimensional feature space to represent visual data, instead of the 105 dimensional original space. In order to train the HMMs, annotations are needed. First the acoustic annotations provided in the AV-LASYN database were used but we quickly came up with the conclusion that annotations based on audio are not suitable for visual modelling. We then tried to annotate manually a subset of the database based on the visual behaviour. Three classes were used: *laugh*, *smile* and *neutral*. The results of the training based on these new annotations gave much better results. Since annotating manually is a highly time consuming task, we have tried to do it automatically, using clustering techniques. Gaussian Mixture Models were used for this purpose. A GMM was fitted to the whole visual data and each frame was classified among 3 clusters based on this fitting. From this classification, we derived visual annotations to be used in the HMM training. The resulting trajectories appeared to be plausible facial laugh motion.

Reactive Visual Laughter Synthesis As we did for audio, we then tried to integrate these models into MAGE to be able to synthesise facial motion reactively. Therefore we had to add a module to MAGE so as to be able to project back the synthesised trajectories into the high dimensional original space. After this projection, the data is available in a format which may be retargeted on a 3D face model. This was done by using Blender in which we loaded an already-rigged 3D face model. Data is sent trough OSC from MAGE to Blender where it is read and applied to the 3D face with a python script. As a proof of concept, we decided to synthesise a succession of neutral and laughing faces in a loop. We also added a control parameter that allows to change the intensity of the visual laughter in realtime. This control parameter amplifies or attenuates the

generated trajectory dynamics. An illustration of the reactive visual laughter in Blender is given in Fig. 5.

4.3 Reactive Exploration of a Stylistic Gait Space

Motion style is something difficult to capture since it is hardly describable. Our human expertise enables us to decode effortlessly the emotion, quality or style conveyed in otherwise functional motions. However it is almost impossible to formally describe the alterations which, once applied to the functional basic motion, give it its specific style. Furthermore, making the distinction between the variability of human motion execution and the style of the motion itself is an additional difficulty when aiming at modelling the style of a motion. Indeed, when performing twice the same motion with the same style, the execution of the movement will always slightly vary. In this work we implemented a framework for stylistic exploration of motion, using the expressive walk case study as a proof-of-concept. Our approach is to foster the generative exploration of styles, from statistical models, as a way of highlighting their implicit properties.

Stylistic Gait Modelling and Synthesis The statistical nature of HMMs enables them to take into account the intrinsic variability of execution of human motion. Both the duration variation and the execution variation are modeled, and a HMM trained on stylistic mocap data becomes a summary of that particular style. Using the Mockey database as training data, the walk was modelled by one five-states left-to-right HMM per step (left and right), following the approach presented in [14]. In a first phase, a global model was trained using all the database. In a second phase, an adaptive training was conducted in order to adapt the generic walk model to each one of the eleven styles present in the database, giving a total of eleven style-specific walk models and one neutral global model. Such an approach corresponds to the left side of Fig. 3, as described in Section 2. In these models, a diagonal covariance matrix is used when modelling the pdfs of the observations, hence not taking into account the interdependency existing between the different body joints motions. Each one of these models can be used to synthesise new walk sequences of any chosen length, and the generated walk sequences will display the style of the models from which they have been generated.

Continuous Stylistic Gait Synthesis However in addition to style and motion variability, the alterations of the functional motions not only convey the style of the motion, but also the intensity of expression of that specific style. Since that intensity can vary continuously, it is impossible to capture the whole range of intensity during motion capture sessions, even for one single style. With our twelve gait models, we are able to generate walk sequences which display the same styles as the ones present in the training database, plus one "neutral" style trained on all the styles. However since all of our models present the same structure, as they have all been adapted from the same generic model, we can

take advantage of this alignment in order to produce new walk styles which have not been recorded. The model parameters space (mean and variances of output and duration pdfs) is considered as a continuous stylistic space, where the values corresponding to each recorded style can be viewed as landmarks which tag the space. Through interpolation and extrapolation between these landmarks, we are able to obtain new walk style models. The intensity of each style can be controlled by interpolating between the original style and the neutral style parameters, also enabling the production of exaggerated and opposite styles. Completely new walk styles can also be built by combining any of the existing styles, enabling the free browsing of the complete stylistic walk space. This approach has been validated in [39]. However in this work both the control of the style and the walk synthesis were implemented as offline processes, preventing the desired free interactive user exploration.

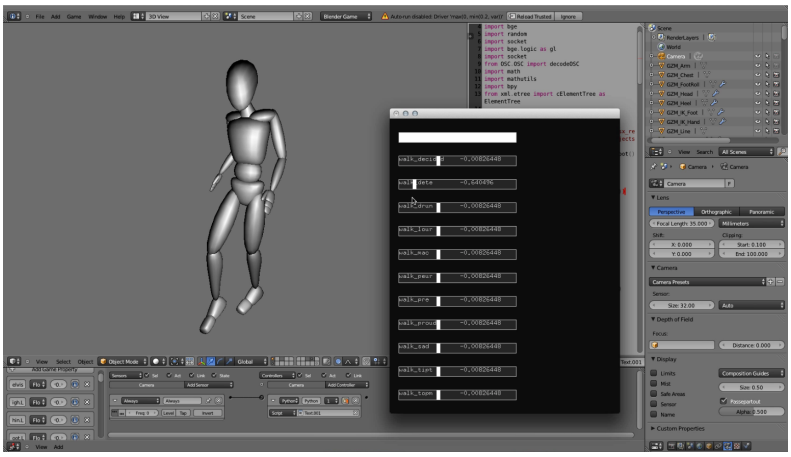

Fig. 6. Illustration of the application for gait style exploration: the MAGE application sends trajectories corresponding to interpolated gait models to Blender through OSC, where the 3D character is animated. The MAGE interface gives one slider per style.

Reactive Stylistic Gait Synthesis In the current work, we implemented a reactive gait style exploration application, enabling the user to reactively control the style of the synthesised walk thanks to MAGE, and to visualise the resulting motion sequence in realtime. In this application, the user browses this stylistic space in realtime, through a set of sliders controlling the influence of each original style, as illustrated in Fig. 6. These stylistic weights are sent to MAGE, which synthesises an infinite walk sequence (a loop of left and right steps), and the walk model is adapted in realtime with the weights corresponding to the sliders, hence modifying the style of the synthesised walk. The synthesised walk trajectories are sent to Blender through OSC, where it is displayed in realtime on a virtual

3D character. The user is hence given interactive control of the walk of a virtual character, as he manipulates sliders to control the style, and can see the influence of these stylistic modifications on the walk of the Blender virtual character. This proof of concept application opens the doors to many possibilities as the size of motion capture databases nowadays explodes and more and more applications seek new possibilities for exploring motion style or compare motions.

5 Realtime HMM-Based Continuous Mapping

The second step in creating our new framework was to validate some mapping strategies within the HMM-based approach – such as described in Section 2. Therefore we have developed a few use case prototypes where the user control was captured and decoded gestures. This Section explains these applications.

5.1 Audio-Visual Face Retargeting

Speaker identity conversion refers to the challenging problem of converting multimodal features between different speakers so that the converted performance of a source speaker can be perceived as belonging to the target speaker. We addressed the problem of speaker conversion using audio and 3D visual information. The speech signal and the 3D scans of a source speaker for a certain utterance will be modified to sound and look as if uttered by a target speaker. The speaker-specific features are mapped between a source and a target speaker using GMMs, as described in Section 2.

In the offline version, the 3D BIWI dataset [40] is used to train the GMM model between any two speakers. The training is done on 40 utterances performed in a neutral manner by both speakers. Spectral features are extracted at a segmental level using the STRAIGHT vocoder [37] which decomposes speech into a spectral envelope without periodic interferences, F0 and relative voice aperiodicity. From the spectral envelope, we use the 1st through 24th Mel-cepstral coefficients, a widely made choice in voice conversion and voice synthesis/analysis systems. The speaker-specific facial articulation features are captured from a dense mesh of 3D data. From the dataset, 7 speech and expressive movement components are extracted following a guided PCA method [41]. As the mouth opening and closing movements have a large influence on face shape, the first jaw component is used as a first predictor, iterative PCA is performed on residual lips values and the next 3 lips components are obtained. The second jaw component is used as the 5th predictor and the last two parameters are extracted as expressive components and represent the zygotic and eyebrow muscle movements. These features are computed at the original video frame rate and are later oversampled to match the audio frame rate. Both visual and spectral features are concatenated with their first derivatives in order to be used for the MLE-based mapping approach.

With the purpose of creating an interactive scene in which virtual actors are able to interact in realtime as guided by a director, we are looking into the possibility of a realtime conversion framework. For this framework, a new system

setup is used, involving a Primesense Carmine 1.09 camera to capture close-range face expressions and a microphone for audio signal acquisition. Also new approaches are needed for extracting relevant audio-visual features. Therefore the Faceshift software [43] is used to generate a realtime 3D face mesh while a speaker performs in front of the camera. For each frame, 48 parameters that control different face movements (jaw opening, eye squint, puff, sneer etc) are also generated. They are called *blendshapes* and can be used with the associated mesh of the user or retargeted to an existing mesh. In the case of audio features, a realtime version of the SPTK tool and MLSA filter are implemented for extracting MFCC coefficients and audio synthesis.

The GMM models are trained offline on a database composed of recorded audio signals of two speakers and the associated blendshapes generated from Faceshift. The converted blendshapes can also be sent to Blender to create a realtime face animation using the 3D mesh of the target speaker. The communication between the different softwares in realtime is done in Max. Like the SPTK and MLSA realtime tools, GMMmap is a module for gaussian mixture model regression using the MMSE method implemented in Max. It uses the models that were trained offline and saved in a suitable format and the audio-visual features that are extracted in realtime to estimate the converted features.

5.2 Realtime Stylistic Full-Body Gait Reconstruction

In our application for exploring the stylistic gait space described in Section 4.3, the ongoing motion is created by the linear combination of the twelve distinct stylistic walk models, according to the weights given to each style on the GUI. Such an approach requires that the stylistic space is explicitly tagged according to the names used in the training database and proposes a vision of the continuum between styles based on linear interpolation.

However considering that the various styles can be named and interpolated within the feature space is a strong design decision. Many use cases might benefit from more implicit approaches towards stylistic exploration. Particularly we wanted to give the user the ability explore the stylistic space through HMM-based mapping between his/her input gestures and the corresponding output. With this idea, we refer to the right side of Fig. 3 where mapping plays the important role in browsing the feature space, as described in Section 2.

In order to validate that the inherent style of a motion can be determined from a subset of its dimensions and remapped in realtime on the remaining dimensions, we have built a prototype that will reconstruct the gait (step sequence plus style) from the upper to the lower body. It means that each 54-channel (18 nodes, each with 3 angles) feature vector from the Mockey database is actually split into inputs and outputs. We consider that the 36 channels corresponding to the upper body (from head to hips) are inputs. The other 18 channels corresponding to leg joints are considered as outputs. They will be animated in realtime by the system. The whole process is illustrated in Fig. 7.

To achieve the regression between upper and lower body dimensions, we implemented a HMM-based mapping as explained in Section 2. The sequence

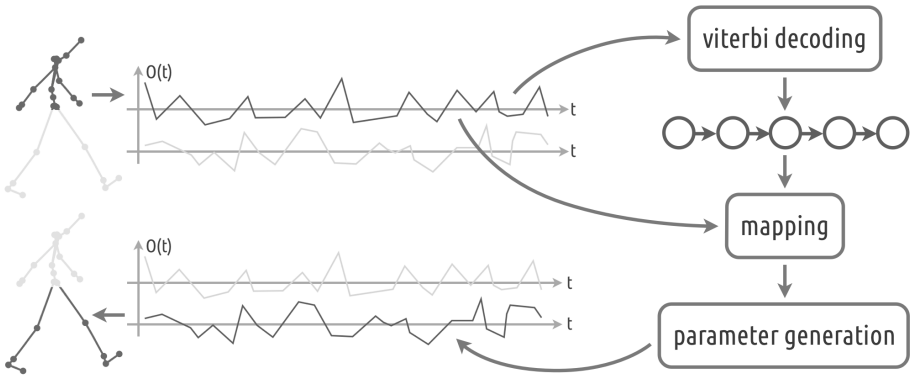

Fig. 7. Illustration of the overall process used in the gait reconstruction example: continuous inputs are decoded with a realtime Viterbi algorithm. This decoding generates a ongoing state sequence that is used for synthesis of the outputs. Before pdfs are used for synthesis, means are modified by a mapping function based on covariance.

of inputs $\mathbf{x}$ are the channels of the upper body and the sequence of target feature vectors $\hat{\mathbf{y}}$ to be estimated are the channels of the lower body. The gait models used are trained on all the styles with full covariance matrices in the pdf representation of the observations. We have a HMM for the right step and a HMM for the left step. Each HMM owns four states.

The first stage in this process is the decoding of the input sequences. The implemented solution for the decoding uses the HTK software toolkit [45]. A realtime data stream is simulated by sending the input data with a specified frame rate to the HRec algorithm, a HTK module that applies the Viterbi algorithm. We added, in the pdfs of the observations, a mask vector to inhibit the channels corresponding to the outputs of the mapping process. This decoding stage provides the most likely HMM $\hat{\lambda}$ that is being played by the streamed data and the current state for this model. To ensure that this stage works in realtime, we extract the partial results given by the Viterbi algorithm without being sure to have the realisation of the complete states sequence for a given model. Moreover, for a given frame $\mathbf{x}_t$, only its past $[\mathbf{x}_1, .., \mathbf{x}_{t-1}, \mathbf{x}_t]$ is taken into account to establish the current state $\mathbf{q_t}$. It appears that it could be more accurate to compute this state by introducing a delay in order to get some information about the future of each sample to choose the best states sequence.

Once the decoded state is available, it can be used to query the HMM database of the upper body dimensions so as to build the state sequence for the synthesis stage. Before the stack of pdfs is accumulated for synthesis, the means of each state are extracted and corrected according to the mapping function described in Section 2. This process tends to influence the means so as to move within the model and react to the covariance information which is expressed between the input and output dimensions. As a result, the statistical properties of the state sequence get modified. When this modified state sequence enters the

synthesis stage, it reflects the stylistic influence of the inputs on the outputs. It means that the style of the upper body transfers to lower body trajectories.

6 Architecture and Software

Based on the reactive properties of HMM-based speech synthesis framework, as described in [26], we built a new speech synthesis library, called MAGE [23]. MAGE is based on the HMM-based parametric speech synthesis system (HTS), which it extends in order to support realtime architecture and multithreaded control. As it is based on HTS, it inherits its features, advantages and drawbacks [26]. The contribution of MAGE is that it opens the enclosed processing loop of the conventional system and allows reactive user control over all the production levels. Moreover, it provides a simple C++ API, allowing reactive HMM-based speech synthesis to be easily integrated into realtime frameworks [46, 47].

6.1 Threaded Architecture of MAGE

One important feature of MAGE is that it uses multiple threads, and each thread can be affected by the user which allows accurate and precise control over the different production levels of the artificial speech. As illustrated in Fig. 8, MAGE integrates three main threads: the *label thread*, the *parameter generation thread* and the *audio generation thread*. Four queues are shared between threads: the *label queue*, the *model queue*, the *parameter queue* and the *sample queue*.

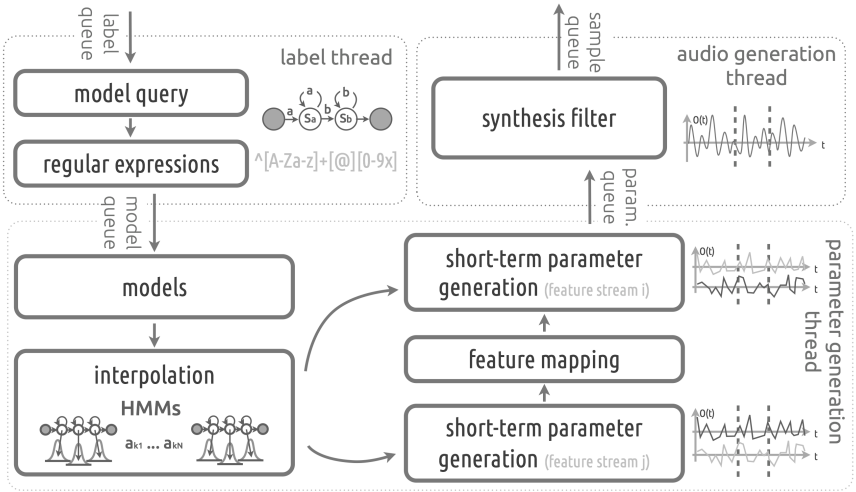

Fig. 8. MAGE: reactive parameter generation using multiple threads and shared queues

The *label thread* controls the phonetic labels, by pulling the targeted phonetic labels from the *label queue* and pushing the corresponding models into the *model*

queue. It is responsible for the contextual control of the system. The *parameter generation thread* reads from the *model queue* a model that corresponds to one phonetic label at a time. For that single label / model the speech parameters are generated (static and dynamic features), which are locally-maximised using only the current phonetic label / model (and if available, the two previous labels). In other words, for every single input phonetic label, the feature vectors are estimated by taking into account the HMMs of that specific label. The generated speech parameters are stored in the *parameter queue*. Finally, the *audio generation thread* generates the actual speech samples corresponding to the inputed phonetic label and store them in the *sample queue* so that the system's *audio thread* will access them and deliver them to the user. Further details of the MAGE reactive parameter estimation can be found in [49].

6.2 Reactive Controls

Accessing and controlling every thread has a different impact over the synthesised speech, as illustrated in Fig. 9. The *label thread* can provide contextual phoneme control. Indeed, the context of the targeted output can be easily manipulated in realtime by simply controlling which of the available phonemes for processing will be inputed into the system and in which sequence.

The *parameter generation thread* can reactively modify the way the available models are used for the parameter sequence generation [49]. The user can reactively alternate between the available models, or interpolate between models with different interpolation weights among the various feature streams. It is also possible to manipulate the mapping between different feature streams, i.e. how a given stream influences another [17].

Finally the *audio generation thread* manipulates reactively the vocoding of every sample, resulting in prosody and voice quality controls. The delay in applying any received user control varies between a single speech sample and a phonetic label depending on the thread that is being accessed. For every thread it is also possible to enable the logging functionality, described hereafter, to store the applied user controls as well as its generated output.

6.3 Reactive Control Through Regular Expressions

One new feature added in MAGE is the support of regular expressions. As explained in [26], in order to describe a phoneme, additional linguistic specifications have to be taken into account. Therefore, every phonetic label in addition to phoneme information, uses various linguistic contexts such as lexical stress, pitch accent, tone, and part-of-speech information for the context-dependent modelling. An example of the format of the phonetic labels can be found in [13].

Until now, it was possible to control the sequence of the labels inputed to MAGE be synthesised. However, there is need for more accurate and specific control over the targeted context. In order to achieve that we use regular expressions that describe the phonetic labels. The integration of regular expressions

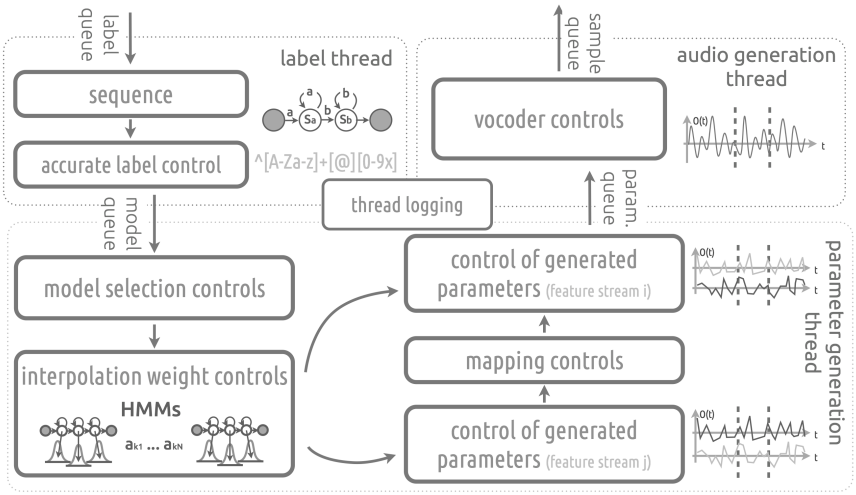

Fig. 9. MAGE: reactive user controls over the multiple threads

"allows the user to query" every imputed label and accordingly to apply certain controls, on every production level of the artificial speech.

For example, when it comes to the contextual control, that occurs in the *label thread*, if the current phoneme is " ɒ " the synthesis of that phoneme can be skipped. Another example, while controlling the models themselves through the regular expressions, a control that occurs respectively at the *parameter generation thread*, if the next phoneme is " ə " we want to interpolate speaking style i with speaking style j using interpolation weight vector $\boldsymbol{y}$. Finally, while controlling the actual generated speech samples, accessing the *audio generation thread*, if the phoneme is stressed then the pitch can be shifted, etc.

6.4 Reactive Mapping Control

In previous versions of MAGE, the granularity of the controls that the user can access for the parameter generation stopped at the model level. Indeed, through the API, a user could only push left-to-right HMMs into the model queue and, from there, compute the duration of each state and the sequence of corresponding observations. This constrains the use case into a left-to-right pattern that does not allow integration of the mapping control detailed in Section 5. Therefore, we added a state queue into MAGE as an alternative to the model queue.

While the model queue is usually filled with sequences of states corresponding to models selected to match the labels in the label queue, the state queue is fed directly with one state at a time. Each state corresponds to one frame of observations and, as such, has a duration of one. If the system must remain in a state for N frames, that state is simply pushed in the queue N times. This enables arbitrary patterns and number of states for the HMMs and thus overcomes the limiting effect of the model queue.

As for the short-term computation of observation frames from the state queue, it is achieved almost as for the model queue. The most significant difference is that one can set M, the number of frames to be computed whereas in the model queue the frames are computed for one complete model at a time, and the number of frames generated is equal to the duration of that model. The context for the short-term parameter computation is set in the same fashion as for the model queue, except that the user sets a number of states to be considered before and after instead of a number of models. This notably allows to always use a constant amount of contextual information, for instance 17 states in the past and 3 states in the future of the M states that correspond to the M frames to be computed. This contrasts with the model queue for which the amount of contextual information is the sum of the durations of each model used as context and thus can change at every step. Using the state queue with $M = 1$, one can even make the computation for one state at a time. In other words, one can generate one frame at a time, while still using surrounding states as the context for the short-term parameter computation.

6.5 Logging of User Actions and Generated Output

One of the major complications when working with reactive applications is that of detecting and explaining unexpected situations. Indeed, every action from the user can cause an instant, or not so instant, reaction from the system. Depending on many factors (which MAGE's thread it is applied to, OS process scheduling policy, etc.), both the reaction and its time delay after the action occurs can be different, even if the user reproduces the same set of interactions. Therefore, when something unusual happens it can be very difficult to, first, realise it and then reproduce it to eventually pinpoint the cause of the event. It could simply be a normal, albeit surprising, answer to a one in a million combination of user commands but it might as well be a bug in the application or in the core MAGE library. Added to this is the problem of detecting exactly when it happened.

In order to make it easier to solve these issues, we introduced a simple logging system in MAGE, as illustrated in Fig. 9. If enabled, it records the sequences of controls sent to MAGE by the user such as the labels, pitch, α, interpolation weights, etc. Each of these values is recorded with a timestamp corresponding to the index of the first sample to which it is actually applied inside of MAGE. Besides, the logging system also saves the evolution of the inner state of MAGE. This is currently limited to the content of the frame and sample queues but could easily be extended to the model and state ones.

7 Conclusions

In this project we have gathered different approaches and backgrounds, with the common aspect of being interested in applied statistical modelling, and we have created a unified framework. This framework is based on trajectory GMMs and HMMs and use a newly-created gesture recognition tool and a new version

of MAGE in order to enable the development of mapping strategies. The idea of HMM-based mapping has been formalised and generalised to any kinds of input and target stream of features. Such a reflexion had a significant impact in how we were envisioning the use of generative models in this work. Therefore we have been able to create a first set of new prototype applications to assess our approach of parameter generation. Indeed we have created an incremental speech synthesiser, generating speech audio right when the user is typing with a limited delay. The current incremental synthesis system allows for a simple analysis relying on a lexical look-up to provide simple lexical analysis and word prediction. This sufficiently demonstrates the possibility of realtime incremental TTS. Many possible future directions can be perceived, such as the implementation of better user interfaces for faster input to the system, the utilisation of MAGE's realtime speech modifications capabilities (e.g. to adjust synthesis speed to user input speed) and the prediction of future phones (and other contexts) to allow the system to be truly realtime without a significant loss of synthesis quality. Also in the parameter generation improvements, we have created realtime exploration of mocap-based trajectories. This idea has been applied to face and body. For the face, it gave the first realtime audio and visual laughter prototype. For the body, we created a new application for exploring the stylistics of gait by blending together various identified one-style models and enable inhibition and exaggeration of those styles. The second step in our work has been to design and assess more implicit statistical mapping applications, where the input is a natural gesture. There we have developed a audio-visual speaker retargeting prototype, where the expressive multimodal speech gestures of a given speaker are remapped on another one in realtime. Also we have created the first gait reconstruction application, where the upper body gait (balancing arms and torso) triggers the parameter generation corresponding to the lower body motion (legs) in realtime. This prototype demonstrates that that HMM-based recognition of stylistic data, its mapping and the corresponding parameter generation can be achieved in a realtime scenario. Finally most of these developments have helped the MAGE software to head towards its third major release.

Acknowledgement

N. d'Alessandro is funded by a regional fund called Région Wallonne FIRST Spin-Off. J. Tilmanne and T. Ravet are supported by the European Union, 7th Framework Programme (FP7-ICT-2011-9), under grant agreement n^o 600676 (i-Treasures project). M. Astrinaki and O. Babacan are supported by a PhD grant funded by UMONS and Acapela Group. H. Çakmak receives a PhD grant from the Fonds de la Recherche pour l'Industrie et l'Agriculture (FRIA), Belgium. The work of Adela Barbulescu has been partially supported by the LabEx PERSYVAL-Lab (ANR-11-LABX-0025)

References

1. Mori, M.: The Uncanny Valley. Energy, vol. 7, no. 4, pp. 33–35, 1970.

2. Mori, M.: The Uncanny Valley (K. F. MacDorman & N. Kageki, Trans.). IEEE Robotics & Automation Magazine, vol. 19, no. 2, pp. 98–100, 2012.
3. Dutoit, T.: An Introduction to Text-To-Speech Synthesis. Kluwer Academic Publishers Inc., 1997.
4. Raux, A., Black, A. W.: A Unit Selection Approach to F0 Modelling and its Applications to Emphasis. In: IEEE Workshop on Automatic Speech Recognition and Understanding, pp. 700–705, December 2003.
5. Lindemann, E.: Music Synthesis with Reconstructive Phrase Modelling. IEEE Signal Processing Magazine, pp. 80–91, vol. 24, no. 2, March 2007.
6. Fechteler, P., Eisert, P., Rurainsky, J.: Fast and High Resolution 3D Face Scanning. In: IEEE International Conference on Image Processing, vol. 3, pp. 81–84, 2007.
7. Menache, A.: Understanding Motion Capture for Computer Animation and Video Games. Morgan Kauffman Publishers Inc., 2000.
8. d'Alessandro, N.: Realtime and Accurate Musical Control of Expression in Voice Synthesis. PhD defence at the University of Mons, November 2009.
9. Maestre, E., Blaauw, M., Bonada, J., Guaus, E., Perez, A.: Statistical Modelling of Bowing Control Applied to Violin Sound Synthesis. IEEE Transactions on Audio, Speech, and Language Processing, vol. 18, no. 4, pp. 855–871, 2010.
10. Tokuda, K., Yoshimura, T., Masuko, T., Kobayashi, T., Kitamura, T.: Speech Parameter Generation Algorithms for HMM-Based Speech Synthesis. In: Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP'00), vol. 3, pp. 1315–1318, 2000.
11. Dutreve, L., Meyer, A., Bouakaz, S.: Feature Points Based Facial Animation Retargeting. In: Proceedings of the 2008 ACM Symposium on Virtual Reality Software and Technology, pp. 197–200, 2008.
12. Hunt, A., Wanderley, M., Paradis, M.: The Importance of Parameter Mapping in Electronic Instrument Design. Journal of New Music Research, vol. 32, no. 4, pp. 429–440, 2003.
13. Tokuda, K., Oura, K., Hashimoto, K., Shiota, S., Takaki, S., Zen, H., Yamagishi, J., Toda, T., Nose, T., Sako, S., Black, A. W.: HMM-based Speech Synthesis System (HTS), `http://hts.sp.nitech.ac.jp`
14. Tilmanne, J., Moinet, A., Dutoit, T.: Stylistic Gait Synthesis Based on Hidden Markov Models. Eurasip Journal on Advances in Signal Processing, vol. 2012(1), no. 72, 2012.
15. Urbain, J., Cakmak, H., Dutoit, T.: Evaluation of HMM-Based Laughter Synthesis. In: Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP'13), pp. 7835–7839, 2013.
16. Astrinaki, M., d'Alessandro, N., Picart, B., Drugman, T., Dutoit, T.: Reactive and Continuous Control of HMM-based Speech Synthesis. In: IEEE Workshop on Spoken Language Technology, December, 2012.
17. Astrinaki, M., Moinet, A., Yamagishi, J., Richmond, K., Ling, Z.-H., King, S., Dutoit, T.: MAGE - Reactive Articulatory Feature Control of HMM-Based Parametric Speech Synthesis. In: Proceedings of the 8th ISCA Speech Synthesis Workshop (SSW8), September, 2013.
18. Hueber, T., Bailly, G., Denby, B.: Continuous Articulatory-to-Acoustic Mapping using Phone-Based Trajectory HMM for a Silent Speech Interface. In: Proceedings of Interspeech, ISCA, 2012.
19. Kay, S. M.: Fundamentals of Statistical Signal Processing Volume 2: Detection Theory. Prentice Hall PTR, 1998.

20. Stylianou, Y., Cappé, O., Moulines, E.: Continuous Probabilistic Transform for Voice Conversion. IEEE Transactions onSpeech and Audio Processing, Vol. 6(12), pp. 131–142, IEEE (1998).
21. Kain, A. B.: High Resolution Voice Transformation. PhD Thesis, Rockford College, 2001.
22. Toda, T., Black, A. W, Tokuda, K.: Voice Conversion Based on Maximum-Likelihood Estimation of Spectral Parameter Trajectory. In: IEEE Transactions on Audio, Speech, and Language Processing, vol. 15, no. 8, pp. 2222–2235, 2007.
23. Astrinaki, M., Moinet, A., Wilfart, G., d'Alessandro, N., Dutoit, T.: MAGE Platform for Performative Speech Synthesis, `http://mage.numediart.org`
24. Kominek, J., Black, A. W.: CMU Arctic Databases for Speech Synthesis. Tech. Rep., Language Technologies Institute, School of Computer Science, Carnegie Mellon University, 2003.
25. Imai, S., Sumita, K., Furuichi, C.: Mel Log Spectrum Approximation (MLSA) Filter for Speech Synthesis. Electronics and Communications in Japan, part I, vol. 66, no. 2, pp. 10–18, 1983.
26. Tokuda, K., Nankaku, Y., Toda, T., Zen, H., Yamagishi, J., Oura, K.: Speech Synthesis Based on Hidden Markov Models. In: Proceedings of IEEE, vol. 101, no. 5, 2013.
27. Sundberg, J.: The Science of Singing Voice. PhD Thesis, Illinois University Press, 1987.
28. Titze, I. R.: Nonlinear Source-Filter Coupling in Phonation: Theory. J. Acoust. Soc. Am., vol. 123, pp. 2733–2749, 2008.
29. Babacan, O., Drugman, T., d'Alessandro, N., Henrich, N., Dutoit, T.: A Comparative Study of Pitch Extraction Algorithms on a Large Variety of Singing Sounds. In: Proceedings of ICASSP, 2013.
30. Babacan, O., Drugman, T., d'Alessandro, N., Henrich, N., Dutoit, T.: A Quantitative Comparison of Glottal Closure Instant Estimation Algorithms on a Large Variety of Singing Sounds. In: Proceedings of ICASSP, 2013.
31. Tilmanne J., Ravet, T.: The Mockey Database, `http://tcts.fpms.ac.be/~tilmanne`
32. IGS-190, Animazoo website, `http://www.animazoo.com`
33. Baumann, T., Schlangen, D.: Recent Advances in Incremental Spoken Language Processing. In: Interspeech 2013 Tutorial 1, 2013.
34. Oura, K.: An Example of Context-Dependent Label Format for HMM-Based Speech Synthesis in English. In: HTS-demo_CMU-ARCTIC-SLT, from `http://hts.sp.nitech.ac.jp`, 2011.
35. Urbain, J., Cakmak, H., Dutoit, T.: Evaluation of HMM-based Laughter Synthesis. In: IEEE International Conference on Acoustics Speech and Signal Processing (ICASSP), 2013.
36. Urbain, J., Cakmak, H., Dutoit, T.: Automatic Phonetic Transcription of Laughter and its Application to Laughter Synthesis. In: Proceedings of the 5th biannual Humaine Association Conference on Affective Computing and Intelligent Interaction, 2013.
37. Kawahara, H.: Straight, Exploitation of the Other Aspect of Vocoder: Perceptually Isomorphic Decomposition of Speech Sounds. Acoustical Science and Technology, vol. 27, no. 6, 2006.
38. Drugman, T., Wilfart, G., Dutoit, T.: A Deterministic Plus Stochastic Model of the Residual Signal for Improved Parametric Speech Synthesis. In: Proceedings of Interspeech, 2009.

39. Tilmanne, J., Dutoit, T.: Continuous Control of Style and Style Transitions through Linear Interpolation in Hidden Markov Model Based Walk Synthesis. Transactions on Computational Science XVI, vol. 7380, pp. 34–54, 2012.
40. Fanelli, G., Gall, J., Romsdorfer, H., Weise, T., Van Gool, L.: Acquisition of a 3D Audio-Visual Corpus of Affective Speech. IEEE Transactions on Multimedia, vol. 12, no. 6, pp. 591–598, 2010.
41. Bailly, G., Govokhina, 0., Elisei, F., Breton, G.: Lip-Synching Using Speaker-Specific Articulation, Shape and Appearance Models. EURASIP Journal on Audio, Speech, and Music Processing, vol. 2009, no. 5, 2009.
42. Barbulescu, A., Hueber, T., Bailly, G., Ronfard, R.: Audio-Visual Speaker Conversion Using Prosody Features. In: International Conference on Auditory-Visual Speech Processing, 2013.
43. Faceshift, `http://faceshift.com`
44. Max Audio Software, `http://cycling74.com/products/max`
45. University of Cambridge, The Hidden Markov Model Toolkit (HTK), `http://htk.eng.cam.ac.uk`.
46. Puckette, M., Pure Data, `http://puredata.info`.
47. Lieberman, Z., Watson, T., Castro, A., et al: openFrameworks, `http://www.openframeworks.cc`.
48. Astrinaki, M., Moinet, A., d´Alessandro, N., Dutoit, T.: Pure Data External for Reactive HMM-based Speech and Singing Synthesis. In: Proceedings of the 16th International Conference on Digital Audio Effects (DAFx-13), September, 2013.
49. Astrinaki, M., d'Alessandro, N., Reboursiere, L., Moinet, A., Dutoit, T.: MAGE 2.0 : New Features And Its Application In The Development Of A Talking Guitar. In: Proceedings of the 13th International Conference on New Interfaces for Musical Expression (NIME´13), May, 2013.